Having thus described the preferred embodiment, the invention is now claimed to be:

1. A battery powered device comprising:

a disk memory (80);

a means (82) for rotating the disk memory;

a read/write means (84) for at least one of reading and writing streaming data from or to the disk memory;

a buffer memory (86) for storing data read from or to be written to the disk memory;

an energy saving scheduling means (96,100) for monitoring the buffer memory and controlling the disk memory rotating means in accordance with the monitored buffer memory.

2. The device as set forth in claim 1 wherein the scheduling means (100) includes:

(1) a means for monitoring how full the buffer memory is; and

(2) a means for monitoring a rate of data transfer into/out of the buffer memory.

3. The device as set forth in claim 2 wherein the scheduling means (100) adjusts scheduled disk starting and stopping times in accordance with the monitored fullness of the buffer memory (86) and the data transfer rate.

4. The device as set forth in claim 3 further including:

a Back-Front-Back scheduling process (100) to reorder a refilling/emtying of various buffers and to remove gaps between buffer memory filling intervals.

5. The device as set forth in claim 1 wherein the reading/writing means (84) both reads and writes and wherein the buffer memory (86) includes a plurality of buffer memories, during a mode with concurrent reading and writing, at least one buffer memory buffers data to be written and at least one buffer memory buffers data that has been read, the scheduling means (100) monitoring the write buffer memory to determine how long before it is full and the read buffer memory to determine how long before it is empty.

6. The device as set forth in claim 1 further including:

a system means (90) for controlling access to the memory means;

a means (92) for user input/output, in communication with the controller means (90);

a means (94) for coherently storing/reading the streams to/from the storage means (80), in communication with the controller means (90).

7. The device as set forth in claim 1 wherein the buffer memory is partitioned into n buffers (80) for n data streams.

8. The device as set forth in claim 7 wherein the scheduling means performs the steps of:

spinning up a disk memory;

filling/emptying each stream i of a set S of the n data streams by reading/writing from/to the storage means until the respective stream is full/empty;

determining an earliest next spinning up time t_0;

putting the storage means in standby mode thereby spinning down the disk memory;

re-determining, at or just prior to time t_0, a new earliest spinning up time t_k based on current buffer fillings $f_i(t)$ at a current time t for each stream i in the set S;

iteratively performing the re-determining the new earliest spinning up time t_k until the time t_k is within a predetermined closeness to t_{k-1} or within a predetermined closeness to the current time t; and,

waiting until time t_k, or just prior to time t_k.

9. The device as set forth in claim 8, wherein the earliest next spinning up time t_0 is within at least $M / \sum_{j \in S} r_j^{\max}$ time units of the current time t, $r_i^{\max}$ being a maximum bit rate of the stream i, and m being a total amount of partitioned memory of the n buffers.

10. The device as set forth in claim 9, further including:

a means for examining the latest possible filling/emptying interval, $[s_i, e_i)$, for each stream i of the set S in order of non-increasing end time, s_i being a start time for the interval of stream i and e_i being an end time, including:

a means for advancing a preceding filling/emptying interval $[s_j, e_j)$ by an advance time $e_j - s_j$ if $e_j > s_j$; and,

a means for re-examining the latest possible filling/emptying interval, $[s_i, e_i)$, for each stream i of the set S in order of increasing end time, including:

a means for removing gaps between successive filling/emptying intervals by advancing intervals behind a gap forward in time, a gap

existing whenever $s_{j+1} > e_j$.

11. An energy efficient disk scheduling method comprising:

at least one of reading and writing streaming data from/to a disk memory;

buffering the data that is at least one of read from and written to the disk memory;

monitoring the buffered data; and

scheduling rotation of the disk memory in accordance with the monitored buffered data.

12. The method as set forth in claim 11 wherein the monitoring includes:

(1) monitoring how full a buffer memory is; and

(2) monitoring a rate of data transfer into/out of the buffer memory.

13. The method as set forth in claim 12 wherein the buffer memory (86) includes a plurality of buffer memories, and further including:

both reading and writing, at least one buffer memory buffering data to be written and at least one buffer memory buffering data that has been read;

monitoring the write buffer memory to determine how long before it is full and the read buffer memory to determine how long before it is empty.

14. The method as set forth in claim 11, further including:

allocating a total amount M of memory for buffering stream data;

partitioning the memory M over a set S of n streams, each stream i of the set S being given a partitioned amount of buffer memory; and,

repeatedly performing scheduling.

15. The method as set forth in claim 14 wherein the scheduling includes:

spinning up a disk memory;

filling/emptying each stream i of set S by reading/writing from/to the storage means until the respective stream is full/empty;

determining an earliest next spinning up time t_0;

putting the storage means in standby mode thereby spinning down the storage means;

re-determining, at or just prior to time t_0, a new earliest spinning up time t_k based

on current buffer fillings $f_i(t)$ at a current time t for each stream i in the set S;

iteratively performing the re-determining the new earliest spinning up time t_k until the time t_k is within a predetermined closeness to t_{k-1} or within a predetermined closeness to the current time t; and,

waiting until time t_k, or just prior to time t_k.

16. The method as set forth in claim 15, wherein the partitioned amount of memory for each stream i of the set S is approximately $r_i^{\max} M / \sum_{j \in S} r_j^{\max}$, $r_i^{\max}$ being a maximum bit rate of the stream i.

17. The method as set forth in claim 15, wherein the earliest next spinning up time t_0 is within at least $M / \sum_{j \in S} r_j^{\max}$ time units of the current time t, $r_i^{\max}$ being a maximum bit rate of the stream i.

18. The method as set forth in claim 15, wherein the total amount M of memory is solid-state memory.

19. The method as set forth in claim 15, further including controlling admission of streams to ensure that $\sum_{i \in S} r_i^{\max} < r_{\text{disk}}$, $r_i^{\max}$ being a maximum bit rate of the stream i, and r_{disk} being a rate of the storage means.

20. The method as set forth in claim 15, further including pausing between two successive streams, j and $j+1$ when all subsequent streams $j+1, j+2, \ldots, n$ can be delayed.

21. The method as set forth in claim 15, further including:

examining the latest possible filling/emptying interval, $[s_i, e_i)$, for each stream i of the set S in order of non-increasing end time, s_i being a start time for the interval of stream i and e_i being an end time, including:

advancing a preceding filling/emptying interval $[s_j, e_j)$ by an advance time $e_j - s_j$ if $e_j > s_j$; and,

re-examining the latest possible filling/emptying interval, $[s_i, e_i)$, for each stream i of the set S in order of increasing end time, including:

removing gaps between successive filling/emptying intervals by

advancing intervals behind a gap forward in time, a gap existing whenever $s_{j+1} > e_j$.

22. The method as set forth in claim 21, further including repeatedly applying the examining and re-examining in each of the scheduling cycles.